The formalisation of specifications from specifications written in natural language

Alain-Jérôme Fougères, Philippe Trigano

The formalisation of specifications from specifications written in natural language

Alain-Jérôme Fougères* and Philippe Trigano**

* Centre National d'Etudes des Télécommunications,
2 av. Pierre Marzin, 22307 Lannion France
E-mail: fougeres@lannion.cnet.fr

** HEUDIASYC URA CNRS 817, U.T.C. , 60200 Compiègne
E-mail: trigano@hds.univ-compiegne.fr

Abstract. The activity of specification is becoming considerable; every day an enormous quantity of pages which, for the most part, is written in natural language. For CNET, which carries out studies of the services and equipment of France Telecom and which has the capability of putting into practice the stages of specification and validation, the need to reduce the time needed in the development of the services is a priority. One method towards achieving this objective is to formalise the maximum number of specifications received. With this in mind, we will try to demonstrate the possibility of a certain automation in the passage from the informal to the formal, by means of methods and proven tools, available to assist an expert in specifications. For this end we propose a process of formalisation which relies on an intermediary representation of the specifications with the formalism of conceptual graphs before arriving at a formal description in Z of the initial spécification.

Key Words : knowledge representation, natural language, conceptual graphs, formal description, formal specifications, Z language.

1 Presentation

1.1 The context of study

This research is motivated, from within CNET, by desire to find methods of reducing the time needed to develop new services in telecommunications. It serves as a means to shorten all the stages in the cycle of development of each service offered (market studies, functional specifications, industrial development, validation, servicing, application and maintenance). In this context, mastering the stage of functional specifications becomes of prime importance, since it facilitates the realisation of the service. Thus, each step which facilitates the writing of the specifications, whilst keeping the expected quality [5], contributes to a reduction of the time, inherent in this stage and, consequently in subsequent stages.

Our objective consists of helping the specification writers of France Telecom to formalise their specifications, by concentrating our attentions on the semantic aspects [8] of an informal specification. The *point de départ* for this procedure of formalisation arrives from specifications written in natural language, comparable to the expression of requirements or to the order book, being converted in the terminology of software science with regard to the target, that must be determined within the context of the not inconsiderable set of formal languages[1].

1.2 Towards a representation in the form of a conceptual graph

There is a general concensus which states that you cannot reasonably envisage passing directly from natural language to formal representation; in the main this being due to the problems inherent in the use of natural language and, more particulary, in its interpretation (ambiguities, context, completude). This has led us to select for the construction of an intermediary semantic representation, defined by Sowa [6]: the model of *conceptual graphs*[1].

This choice is motivated by the existence of numerous works on the representation of texts written in natural language with the aid of conceptual graphs, being transformed directly from the conceptual graphs into a logical formula of the fisrt order and through the work already done on the logical interpretation of these graphs - including the extensions towards logical models of the second order and modals.

1.3 Experimentation

Our experiments are based on the specification of *NEF*[2] (worked out at CNET) and more specically on the tenth chapter detailing the *tarification* (the entire work comprises 2 volumes, about 800 pages). It can evidently be seen that the extensive nature of this specification and the linguistic complexities attached to it, have not permitted us, in this first approach to foresee a complete definitive path from the informal description to a formal specification. We therefore settled, as an experimental protocol, on the realisation of the complete procedure of formalisation in an incremental way.

1.4 The procedure of formalisation

In setting out on this move towars formalisation, we propose a sequence of processes from initial informal specifications, likely to provide us with a formal description (Figure 1).

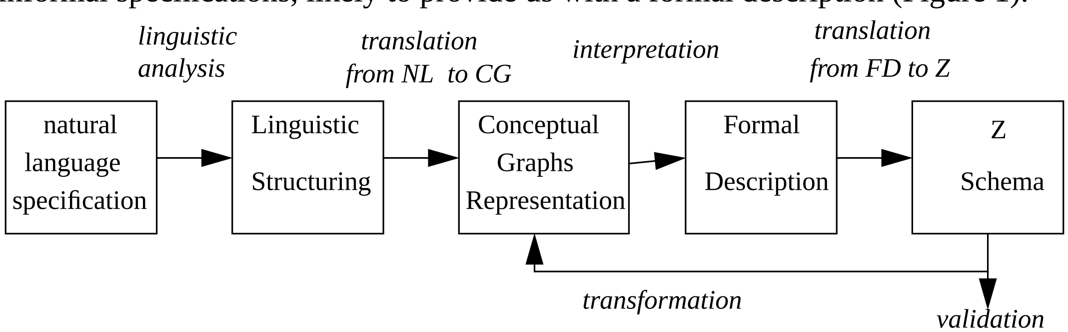

Figure 1. Different stages in the processing of specifications.

2 Description of the work

2.1 Linguistic aspects

The processing of the language breaks down into two stages: a preliminary stage in which there is the acquisition of knowledge pertaining to the domain, and then the actual stage of linguistic analysis, itself. This second stage is generally sub-divised into five stages of analysis morphological, lexical, syntactic, semantic, pragmatic.

2.1.1 The acquisition of knowledge phase

This preliminary stage consists of extracting lexical information contained within the text, in order to determine the preferred links that the words have between them. A simple study of the co-occurrence of words, based on an analysis of lexical proximity, thus enables us to reveal the presence of compound words, of expression, of predicative relationship and even of schemas of phrases peculiar to the domain in question. The united use of this frequential analysis with techniques of statistical filtrage, such as the mutual information[3] permits refinement and

1. More precisely, the languages of formal description, standardized by ISO and CCITT (Estelle, Lotos, SDL) as well as Z language.
1. The recent development in the formalisation of conceptual graphs has led to a standardization (comité X3T2 of ANSI).
2. NEF : *Normes d'Exploitation et de Fonctionnement* (Rules of conduct and operation) of France Telecom.

improvement in the pertinence of the obtained results.

A second phase of knowledge acquisition specific to our modelisation consists of extracting from the dictionary some definitions of terms retained as concepts, in order to describe them in a semantic dictionary in the form of conceptual graphs. In order to automatize this task, we have adapted the algoritms of [3] which allow us to detect the hyperonymic relationship contained in the dictionary definitions and to adjust them, in the eventual presence of terms modifying the definition. Once the content of the definition has been analysed, it is then possible to construct the corresponding conceptual graphs and to include it in a canonical base.

2.1.2 The linguistic analysis phase

The morpho-lexical analysis. In the course of this phase, it is a question of sequencing the analysed sentences in order to obtain a series of words after having identified the simple words, the compound words and the set expressions. This analysis is made possible by the use of a lexicon, like DELAS[1], and by the use of a module of flexional analysis for the description of grammatical forms (conjugation, gender, number). In addition to the recognition of the words of the language, this stage allows the distribution of the words into class (syntactical categories) such as noun, verb, adjective.

The syntactic analysis. The strategies for syntactic analysis are very numerous: analysis by components, formal grammars, grammatical labelling according to probabilities, etc. Our aim here is not to set out the full array of the methods but rather to set out clearly the formalism LFG [4] that we have chosen.

Lexical-Functional Grammars (LFG) break down into two levels :

. the c-structure (analysis by components) described as the method for producing rules of grammar out of context; it represents the syntactical structures in the form of a tree;

. the f-structure (functional description) comprises pairs of function-value, shows the grammatical functions such as subject, object, etc. One particular function, named *Pred*, puts together the syntactical functions with the semantic roles of the predicate, thus facilitating the later semantic interpretation.

The semantic analysis. The selected formalism of representation of semantic knowledge, being the model for the conceptual graphs, this analysis therefore consists of the semantic translation of the syntactical structure into the form of conceptual graph. For this, we have taken inspiration from the case grammars which determine the different thematic roles taken by the components of a phrase with the help of information acquired about word-order, about prepositions, verbs and context. In other words, the analyser determines the way in which the nominal groups of a phrase are bound to the verbs: the semantic role specifying how an object participates in the description of an action.

2.2 The formalism of conceptual graphs

2.2.1 The elements of formalism

Definition. A conceptual graph is a bipartite graph labelled, directed, finite and connected. The two components forming the nodes of the graph are: the concepts and the relations.

A concept, the base unity of the model, consists of a type label and a referent [<type>:<referent>]. The concept types represent the occurrences of a class of objects. They are regrouped in a hierarchical structure which defines a lattice. All the concept types are thus linked by a relation of partial order.

---

3. When the number of couple of lexical unities observed becomes elevated, we estimates the probabilities of pertinent association by a method of likelihood. The mutual information is given by: $\hat{I}(x,y) = log\ (n_{x,y} / n_x n_y)$, with n_x and n_y the number of occurrences of *x* and *y*, and $n_{x,y}$ the number of occurrences of the couple *(x,y)*.

1. Dictionary of the LADL (University Paris VII), containing approximately 80000 simple words with their categorical references.

The referents have the task of defining precisely the sense of the concept by specifying an occurrence of concept type. They can be of differing natures, notably individual, generic or propositional; in the last instance, the referent is a conceptual graph.

The conceptual relations link two or more concepts: [c1]->(relation)->[c2]. The definition of under-relations is sometimes necessary to afford more finesse in the semantic representation when the relations are of a very general nature. A hierarchy of relations is then established, having the same properties as the lattice of concept types.

Sowa [6] defined four elementary operations on conceptual graphs called *canonical formation rules* which permit easy manipulation and canonical derivation of other graphs: copy, restrict, join, simplify. These operations put into action the mecanisms of generalisation and of specialisation of graphs, by exploiting the hierarchical structuring of concept types. To make these operations acceptable, it is necessary to define a relationship of conformity, allowing us to verify, in the event of a change of type label outside the operation, that the latter remains conforming to the type.

2.2.2 Development of an ontology

For a modelisation of contexts using the formalism of conceptual graphs, the ontology[1] of domain corresponds to the whole of the definitions of concept types and of relations, the whole of the schemas collected and the whole of the procedures which allow us to deduce some formal knowledge from the preceding knowledge. This ontology realises a synthesis of the whole of the usable objects in the model of the conceptual graphs.

2.2.3 Isomorphism CG and logic of the fisrt order

Sowa defined the operator ϕ which makes a formula in the predicates logic of the fisrt order correspond to every conceptual graph: (ϕ **: CG -> formula**). Thus, in the following example, the command *tarification of a communication* which is represented by the graph *u* will have for equivalent the logical formula $\phi(u)$:

u : [COMMAND: tarification]<-(ObjectOf)<-[COMMUNICATION: *].

$$\phi(u) : \exists\, x,\ COMMAND(tarification) \wedge ObjectOf(x,tarification) \wedge COMMUNICATION(x)$$

2.3 Towards a formalisation

2.3.1 A formal description

After having followed the steps of the human expert, it seemed interesting to us to simulate these steps, adapting them to the formalism of the conceptual graphs. The following figure (Figure 2) takes up in a synthetic way, the methodology used by an expert.

1. Checking off of the references:
    . listing the parameters used;
    . listing the base formulas of the domain.
2. Procedure of formalisation:
    . extraction of pertinent phrases (or parts of phrases);
    . establishment of pre and post-conditions;
    . deduction of associated logical formulas:
        - definition of predicates,
        - definition of eventual functions,
        - logical formulas for every phrase.

Figure 2. Elaboration of a formal description by a human expert.

2.3.2 Z language as the target language

A specification in Z [7] is formed by a sequence of paragraphs comprisisng schemas, variables and base types of the specification. To every expression appearing in a specification

1. Definition of the ontology as we understand it: the sum total of defined and possible to define knowledges in a context.

in Z is associated a unique type. This type can be one of three sorts, a whole type, a cartesian produced type or still a schema type. The relations or the functions allow us to combine these three sorts of objects. A schema consists of a signature and of a property on this signature called *predicative part*. A signature is a collection of variables, each one possessing a type. They are created by the declarations and they provide the vocabulary necessary to the mathematical instructions expressed by the predicates. A predicate is the expression of a property which is characterised by the whole of the links for which it is true, in function of the signature.

The variables are of two sorts: the local variables which have a reduced scope on their schema of declaration and the global variable which form the object of a declaration outside the schema, which confers a general accessibility on them. Moreover, the formalism contains three standard *decorations* used in the description of the operations on the datas abstract types: «’» to label the final state of an operation, «?» to label its entries and «!» to label its exits.

2.3.3 The translation of CG into Z

The algorithm provided in the following figure gives the main lines of the translation of a simple conceptual graph into Z. We will essentially retain, in this overall approach of the translation, the two successive processes done on the concepts and on the relations. The referent of a concept becomes an element of the whole, represented by the type label; as for the relation, this is the object of a functional definition..

for all concept *C* **do**
begin \ *C*<*C’* in the lattice of concept types
  translate the concept *[C:ref]* by
  if *ref* is generic(*) **then**
    i. in the declaration part: *C : P C’*
    ii. in the predicate part: ∃ *c : C*
  else if ref is individual (#i) **then**
    i. in the declarative part: *ref : C*
    ii. C makes the object of a general declaration \ ex: inclusion of schema **endif**
  endif
end

for all relation *R* **do**
begin \ *[C1]->(R)->[C2]*
  verify the conformity of the types of *C1* and *C2*
  translate the relation *R* between *C1* and *C2* by
    i. in the declaration part: *R: C1 <-> C2*
    ii. in the predicative part: *(C1,C2)* ∈ *R*
  if several relations **then**
    conjunction of the predicat associated to *R* with the others
  endif
end

Figure 3. Elementary algorithm of the translation from conceptual graphs into Z.

The following figures illustrate the process of formalisation (in french) by presenting the three levels of representation of the specification of a simple transmission of messages between a transmitter and a receiver via a channel of transmission.

Le système à spécifier se compose d’un émetteur, d’un canal et d’un récepteur:

. L’émetteur peut à tout instant émettre soit le message 1, soit le message 2, soit recevoir une indication de perte p de la part du canal. Cependant, après une indicationde perte, il est tenu de réémettre le dernier message qu’il a envoyé.

. Le canal peut recevoir les deux types de messages différents. Dans chaque cas, il peut soit transmettre le message à la sortie, soit retourner une indication de perte à l’émetteur, soit perdre le message sans rien signaler du tout. Le canal ne peut accepter un deuxième message que si le premier n’y est plus (transmis ou perdu).

. Le récepteur peut recevoir autant de fois qu’il veut le message 1, mais s’il reçoit le message 2 il se bloque suite à un problème interne.

```
u1: [systeme: #1]-
      ->(BUT)->[specifier]
     ->(COMPOSE_DE)->{[emetteur: *x],[canal: *x],
                               [recepteur: *x]}\.
 2: (POSS)->[PROPOSITION: [emettre]-
      ->(AGNT)->[emetteur: #2]
      ->(TEMP)->[instant: forall]
      ->(OBJ)->[[[message: message1] | [message: message2]]
                | [[recevoir]-
                      ->(OBJ)->[indication: *]->(REL)->[perte: p]
                       ->(INIT)->[canal: #3]\,]]\,]\.
 3: (OBLG)->[PROPOSITION: [reemettre]-
            ->(AGNT)->[emetteur: #2]
            ->(OBJ)->[message: #5]-
                   ->(CAR)->[dernier]
                   <-(OBJ)<-[PROPOSITION: [(PASS)->[envoyer]-
                                  ->(AGNT)->[emetteur: #2]\,]]
           ->(TEMP)->[indication: *]->(REL)->[perte: *]
           ->(COND)->[PROPOSITION:[[envoyer]-
                                 ->(OBJ)->[message: *]\,]\,]\.
```

Figure 4. Statement of the specification and fragments of the representation in the form of CG.

```
INDICATION ::= ok          MESSAGES ::= m1
                | p                     | m2
ETAT :: = bloque                        | none
        | ouvert

┌ Emetteur ─────────────────────
│ entreeDeC? : INDICATION
│ sortieVersC : MESSAGES
└───────────────────────────────

┌ Canal ────────────────────────
│ entreeDeE? : MESSAGES
│ sortieVersE! : INDICATION
│ sortieVersR : MESSAGES
├───────────────────────
│ entreeDeE? ≠ none
└───────────────────────────────

┌ Récepteur ────────────────────
│ rentreeDeC? : MESSAGES
│ etat : ETAT
└───────────────────────────────

┌ Système ──────────────────────
│ ΔEmetteur
│ ΔCanal
│ ΔRécepteur
├───────────────────────
│ (entreeDeC? = p ∧
│ sortieVersC ≠ none) ⇒ sortieVersC’ = sortieVersC
│ sortieVersC ≠ none ⇒ sortieVersC’ ∈ {m1,m2}
│ sortieVersR = none ⇒ sortieVersR’ = entreeDeE?
│ sortieVersE! = p ⇒ sortieVersR’ = none
│ entreeDeE? = sortieVersC’
│ entreeDeC? = sortieVersE
│ rentreeDeC? = m2 ⇒ etat’ = bloque
│ etat = ouvert ⇒ rentreeDeC? =(sortieVersR ∧
│                  sortieVersR’ = none)
└───────────────────────────────
```

Figure 5. Spécification in Z of the *Transmission.*

3 Conclusion

We have proposed in this article an overall presentation of the procedure of formalisation of informal specifications expressed in natural language. The implementation of this procedure passes through three successive stages:

. the establishment of the ontology appropriate to the domain, then the implementation of the set of the algorithms participating in the semantic representation of the specifications in the form of conceptual graphs;

. the linguistic analysis of the content of the informal specification, in order to construct the conceptual graphs corresponding to the analysed sentences;

. the logical interpretation of the semantic representation and the translation into Z language.

A system, thus conceived, can form an *intelligent* component part of a softwares tool-box. It constitutes the module of formal specifications, by helping the writer in an interactive way. It is not, however our aim to construct the *automatic specifier* [1] to whom one could give the title of the specification and who would subsequently provide the schemas of textual representation, devoid of all ambiguity. The tool must therefore remain at the service of the intelligence and creativity of the specifier.

Références

[1]:R. Balzer, *A 15 Year Perspective on Automatic Programming*, in IEEE Transactions on Software Engineering, vol. SE-11, N° 11, pp. 1257-1268, november 1985.

[2]: E. Davis, *Representations of Commonsense Knowledge*, Morgan Kaufmann Publishers, San Mateo CA 1990.

[3]: P. Hernert, *Un système d’acquisition de définitions basé sur le modèle des graphes conceptuels*, Phd thesis of the University Paris XIII, 1993.

[4]: R. M. Kaplan, J. Bresnan, *Lexical-Functional Grammar: A Formal System for Grammatical Representation*, in J. Bresnan ed., *The Mental Representation of Grammatical Relations*, MIT Press, 1982.

[5]: I. Sommerville, *Software engineering*, 4ème édition, Addison-Wesley, Reading, MA, 1992.

[6]: J. F. Sowa, *Conceptual Structures: Information Processing in Mind and Machine*, Addison-Wesley Publishing Company, Reading, MA, 1984.

[7]: J.M. Spivey, *The Z Notation - A Reference Manual*, Prentice Hall International (UK) Ltd, 1992.

[8]: Y. Toussaint, *Méthodes informatiques et linguistiques pour l’aide a la spécification de logiciel*, PhD thesis of the University Toulouse III, 1992.